Formalization and Proof of Secrecy Properties

Dennis Volpano
Computer Science Department
Naval Postgraduate School
Monterey, CA 93943, USA
volpano@cs.nps.navy.mil

After looking at the security literature, you will find secrecy is formalized in different ways, depending on the application. Applications have threat models that influence our choice of secrecy properties. A property may be reasonable in one context and completely unsatisfactory in another if other threats exist.

The primary goal of this panel is to foster discussion on what sorts of secrecy properties are appropriate for different applications and to investigate what they have in common. We also want to explore what is meant by secrecy in different contexts. Perhaps there is enough overlap among our threat models that we can begin to identify some key secrecy properties for wider application. Currently, secrecy is treated in rather ad hoc ways. With some agreement among calculi for expressing protocols and systems, we might even be able to use one another's proof techniques for proving secrecy!

Four experts were invited as panelists. Two panelists, Riccardo Focardi and Martín Abadi, represent formalizations of secrecy as demanded by secure systems that aim to prohibit various channels, or insecure information flows. More specifically, they represent noninterference-based secrecy. The other two panelists, Cathy Meadows and Jon Millen, represent formalizations of secrecy for protocols based on the Dolev-Yao threat model [2]. Below are some specific questions that were asked of each of the panelists:

1. Secrecy is sometimes formulated as a "safety" property in protocol analysis where one is concerned with whether an intruder learns a specific value (a secret). Such a criterion is inadequate for guaranteeing secure information flow in systems where secrets can always be encoded or transmitted in covert ways. Leaks arising by indirect flows from within a process executing a protocol seem as dangerous as those caused by message exchange with an adversary. This is especially true of crypto protocols whose implementations normally admit cryptanalytic attacks. So why does protocol analysis adopt a different criterion?

2. Is there a secrecy property for protocols and systems? Is it noninterference (NI) based? One key problem is encryption. It blows NI-based formulations apart. How can we cope with it? Do we assume perfect encryption and fiddle with notions of equivalence until we get the "desired effect"? Or do we use techniques that are more sensitive to the computational complexity of compromising secrets?

3. Can we study protocol secrecy within the same framework as that used for information flow in a concurrent setting? If not, why?

4. Suppose Mallory imitates Bob in a key establishment protocol with Alice, to get Alice to accept a key that Mallory knows. Is this a failure of secrecy because Alice incorrectly believes that the key is known only to Bob and herself?

Panelists were asked to try to respond to these questions or provide questions that they feel are more appropriate. Their responses are given in the following sections. Thanks to the panelists for participating.

1. Martín Abadi's Reply

Suppose that we wish to require that a protocol preserve the secrecy of one of its parameters, x. The protocol should not leak any information about x—in other words, the value of x should not interfere with the behavior of the protocol that the environment can observe. The parameter x may denote the identity of one of the participants or the sensitive data that is sent encrypted after a key exchange. In general, we cannot express this secrecy property as a predicate on

[0] Appears in the Proc 12th IEEE CSFW, pp. 92–95.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE
01 JUN 1999

2. REPORT TYPE
N/A

3. DATES COVERED
-

4. TITLE AND SUBTITLE
Formalization and Proof of Secrecy Properties

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Computer Science Department Naval Postgraduate School Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:

a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
unclassified	**unclassified**	**unclassified**	**UU**	**3**	

behaviors. On the other hand, representing the protocol as a process $P(x)$, we may express the secrecy property by saying that $P(M)$ and $P(N)$ are equivalent (or indistinguishable), for all possible values M and N for x. Here we say that two processes P_1 and P_2 are equivalent when no third process Q can distinguish running in parallel with P_1 from running in parallel in P_2. This notion of process equivalence (testing equivalence) has been applied to several classes of processes and with several concepts of distinguishability, sometimes accounting for cryptographic operations.

Approaches based on predicates on behaviors rely on a rather different definition of secrecy, which can be traced back to the influential work of Dolev and Yao. According to that definition, a process preserves the secrecy of a piece of data M if the process never sends M in clear on the network, or anything that would permit the computation of M, even in interaction with an attacker.

Neither definition of secrecy implies the other. The first one concerns a process with a free variable x, while the second one concerns a process and a term with no free variables. With the first definition, we can say that $P(x)$ preserves the secrecy of the value of x even when this value may be a boolean; with the second one, it does not make much sense to talk about a secret boolean. In addition, the first definition rules out implicit information flows, while the second one does not. While the exact relations between the definitions remain unclear, I believe that the first one represents a more compelling criterion, and that the second one is a useful approximation that fits better into some formal frameworks.

2. Riccardo Focardi's Reply

Non-Interference (NI) has been introduced with the aim of formalizing security policies in systems. In particular, given two groups of users A and B, the requirement "A must not interfere with B" basically imposes that what is done by users in A cannot modify in any way the behaviour of the users in B. As a consequence, we obtain that the information which is known by users in A can never be revealed to users belonging to B. This gives us a strong notion of secrecy (in systems). For example, through NI requirements, we can easily formalize a multilevel security policy by requiring that users at a certain confidentiality level do not interfere with users at a lower level.

Indeed, NI is a general concept that can also be profitably applied in other settings, as it simply verifies if someone is able to induce a new (potentially dangerous) behaviour. As an example, NI has already been successfully exploited for the automatic verification of cryptographic protocols. Usually, when we consider a cryptographic protocol, we would like to be guaranteed that no enemy is able to introduce any "undesirable behaviours". This is exactly what NI requires. For example, for secrecy an "undesirable behaviour" is represented by the leaking of (secret) information which is detectable by simply observing the state of the enemy.

This reflects the power and the limitations of the use of NI-based properties in the analysis of security protocols:

- On one hand, the generality of NI makes it possible to detect in the same analysis completely different attacks (e.g., secrecy and authentication). This could increase the probability of finding new attacks, since we do not need to fix in advance the specific security property to be checked;

- On the other hand, this kind of analysis requires an additional effort in identifying which are the "undesirable behaviours", i.e., which of the revealed behaviours are attacks and which are not. However, when it is possible to reveal an attack by observing few well-defined events, e.g., in secrecy analyses, this task becomes trivial.

Finally, NI seems to be a good unifying approach to computer and network security. As a matter of fact, after the underlying model has been enriched (in some way) in order to deal with cryptography, NI can be used to analyze protocol secrecy within the same framework as that used for information flow in systems.

3. Cathy Meadows' Reply

Most of the work that has been done on applying formal methods to cryptographic protocols has relied upon the Dolev-Yao model, in which both intruders and honest participants have access to a finite number of well-defined operations obeying a finite set of algebraic rules. In this model the secrecy problem reduces to the problem of determining whether or not an intruder can learn a specific word by combining the set of operations available to it with the set of operations performed by the legitimate participants in the protocol. This is a very simple notion of secrecy: the intruder either learns the word or doesn't. Thus it avoids dealing with the question of whether or not the intruder can learn information about a word or key that would help in cryptanalysis, whether or not the intruder learns relationships between words (for example the relationship between a message and its sender), and whether a conspirator can encode secret information in the execution of the protocol. On the other hand, the Dolev-Yao model gives the protocol analyst a powerful tool for understanding a wide range of authentication properties that can be guaranteed by a cryptographic protocol.

This is in marked contrast to the notion of secrecy used in information flow, in which it is attempted to determine whether one untrusted process H could pass information to another untrusted process L by determining whether or not H has any effect on the system that is visible to L. Not only is this a much more subtle notion of secrecy than the Dolev-Yao version, relying on knowledge of possible as well as actual behavior, but the trust model is different: in the Dolev-Yao model the holders of secrets are trusted, while in the information flow model the holder of secrets H is untrusted, and it is up to the system to provide the guarantee that H does not reveal information.

Since the models satisfy such different requirements, it is difficult to "defend" one against the other. However, it does make sense to ask how they could be made to work together in a system that must satisfy multilevel security requirements and also engage in authentication protocols. For example, we might want to consider a system in which a subject is trusted to engage properly in a cryptographic protocol, but may or may not be trusted not to leak information via covert channels.

In order to understand how the two notions of security can be made to work together, we may want to look at another notion of security for systems: the type provided by access control policies. In the access control model, as in the Dolev-Yao model, the system consists of a set of principals (subjects), some of whom may be dishonest, a set of objects, and a finite set of operations that may be performed on the objects, such as creation, deletion, and the granting and removing of access rights. As in the Dolev-Yao model, the notion of secrecy is simple; it boils down to determining whether or not a subject can gain read access to an object. And, also as in the Dolev-Yao model, it is possible to use the finite set of operations to model a wide range of access control policies and requirements.

A number of attempts have been made to unify access control models with information-flow type models, with some success. The Bell-Lapadula model was perhaps the first; it foundered on the question of downgrading. The ability to downgrade data is necessary, but it also violates a straightfoward information flow policy, since it is an obvious flow from High to Low. Intransitive noninterference policies attempt to rectify this situation by allowing information to flow only through certain channels, such as downgraders. Other work, such as that of Simon Foley [3], has concentrated on developing a framework that allows complex structures of information flow requirements.

An approach like the above would allow us to integrate cryptographic protocol analysis into the information flow model, but at the possible price of ignoring information flow risks that could arise from deliberate information leakage during the execution of the protocol. For this we might want to look at work that has been done in the cryptographic community to address this very problem - known as "subliminal channels" in this context [1]. However, this work introduces an added expense: the introduction of a trusted warden to verify the absence of subliminal messages. Moreover, most of the existing work is applicable only to zero knowledge protocols, which have seen few practial applications. However, a closer look at this work may give us ideas for applying it to more generally applicable protocols and to integrating it with information flow models.

4. Jon Millen's Reply

It may seem odd that an apparently primitive concept like "secrecy" could be formalized in several different ways within a single application area such as cryptographic protocol analysis. But we already have a precedent for a multiplicity of models of secrecy in the analysis of secure operating systems. Figures 1 and 2 show the analogy between the two subjects. It is suggested that there is a good reason for having more than one model, and that similar reasoning applies to both areas.

The pyramid pictures illustrate the progression from simple, basic policies to a more detailed analysis that focuses on localized subsystems. The more focused analysis requires more detail in the system model. In both environments, a really thorough treatment of secrecy has to bring in Shannon's information theory, which implies probabilistic considerations.

There is a fundamental difficulty in applying noninterference to encryption, namely the fact that changes in plaintext cause changes in ciphertext. In a probabilistic context, one can rephrase secrecy as an inability to distinguish secret information from randomly generated text.

References

[1] Mike Burmester, Yvo Desmedt, Toshiya Itoh, Kouichi Sakurai, Hiroka Shizuya, and Moti Yung, A progress report on subliminal-free channnels, In Ross Anderson, editor, *Information Hiding - First*

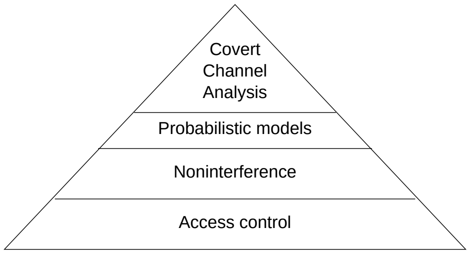

Figure 1. Secure Operating System Analysis

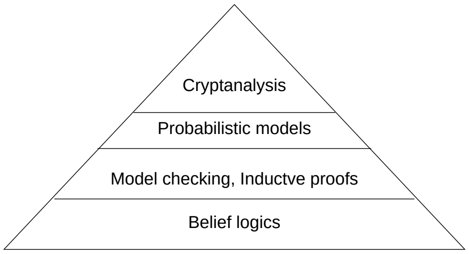

Figure 2. Cryptographic Protocol Analysis

International Workshop, Springer Verlag Lecture Notes in Computer Science 1174, pp. 157–168, June 1996.

[2] Dolev, D. and Yao, A., On the Security of Public Key Protocols, *IEEE Transactions on Information Theory*, 29, 2, pp. 198–208, 1983.

[3] Simon Foley, A taxonomy of information flow policies and models, *Proc 1991 IEEE Computer Society Symposium on Research in Security and Privacy*, IEEE Computer Society Press, pp. 98–108., May 1991.